Discrete and continuous dynamic systems

Petri Nets
Definition and operation

Katalin Hangos

University of Pannonia
Faculty of Information Technology
Department of Electrical Engineering and Information Systems

hangos.katalin@virt.uni-pannon.hu

Apr 2018

Lecture overview

1. Previous notions
   - Discrete event systems
   - Automata models
   - Simple examples
2. Petri net models
   - Description forms
   - Operation (dynamics) of Petri nets
   - Parallel and conflicting execution steps
3. Solution of Petri net models
   - The reachability graph

Discrete event systems

Characteristic properties:

- the *range space* of the signals (input, output, state) is **discrete**: $x(t) \in \mathbf{X} = \{x_0, x_1, ..., x_n\}$
- *event*: the occurrence of change in a discrete value
- *time is also **discrete***: $T = \{t_0, t_1, ..., t_n\} = \{0, 1, ..., n\}$

Only the **order of the events** is considered

- description of sequential and parallel events
- **application area**: scheduling, operational procedures, resource management

Discrete time linear state space models

$$
\begin{aligned}
x(k+1) &= \Phi x(k) + \Gamma u(k) \qquad (\textit{state equation}) \\
y(k) &= Cx(k) + Du(k) \qquad (\textit{output equation})
\end{aligned}
$$

given initial condition $x(0)$;
vector valued signals

$$
x(k) \in \mathcal{R}^n \ , \ y(k) \in \mathcal{R}^p \ , \ u(k) \in \mathcal{R}^r
$$

system parameters:

$$
\Phi \in \mathcal{R}^{n\times n} \ , \ \Gamma \in \mathcal{R}^{n\times r} \ , \ C \in \mathcal{R}^{p\times n} \ , \ D \in \mathcal{R}^{p\times r}
$$

(Not necessarily) equidistant $(t_k - t_{k-1} = \Delta h)$

$$
x(k) = x(t_k) \ \ , \ \ u(k) = u(t_k) \ \ , \ \ y(k) = y(t_k)
$$

Discrete event systems – discrete time state space models

Generalization of discrete time linear state space models

$$
\begin{aligned}
x(k+1) &= \Psi(x(k), u(k)) \qquad (\textit{state equation}) \\
y(k) &= h(x(k), u(k)) \qquad (\textit{output equation})
\end{aligned}
$$

with given initial condition $x(0)$ and nonlinear state Ψ and output function h.

Discrete event system:

1. discrete time with non-equidistant sampling
2. the range space of the signals is discrete
3. event: change in the discrete value of a signal

Automaton - abstract model: $\mathbf{A} = (Q, \Sigma, \delta; \Sigma_O, \varphi)$

- **Set of states**: Q
- **finite alphabet** of the input tape: $\Sigma = \{\#; a, b, ...\}$
- **State transition function**: $\delta : Q \times \Sigma \to Q$
- *Set of initial and final states*: $Q_I, \ Q_F \ \subseteq \ Q$
- **finite alphabet** of the output tape: $\Sigma_O = \{\#; \alpha, \beta, ...\}$
- **Output function**: $\varphi : Q \to \Sigma_O$

Graphical description: weighted directed graph

- **Vertices**: states (Q)
- **Edges**: state transitions (δ)
- **Edge weights**: input symbols (Σ)

Operation of automata

Given

- Initial state: $q_0 \in Q_I \subseteq Q$
- The content of the input tape: $S = [\sigma_1, \sigma_2, ..., \sigma_n]$, $\sigma_i \in \Sigma$

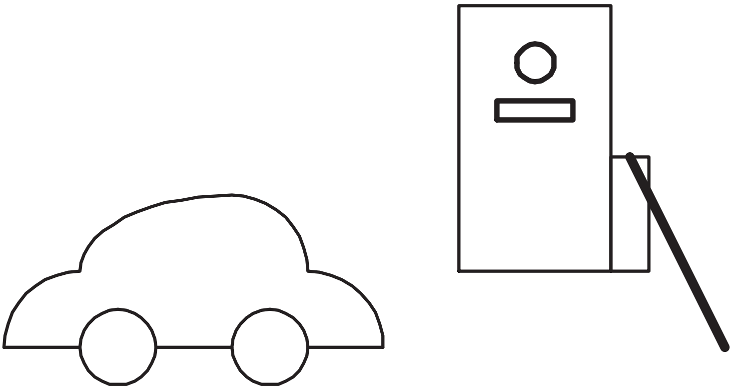
Compute

- Final state: if $q_f \in Q_F \subseteq Q$, then the automaton **accepts** the input
- The content of the output state: $S_O = [\zeta_1, \zeta_2, ..., \zeta_n]$, $\zeta_i \in \Sigma_O$

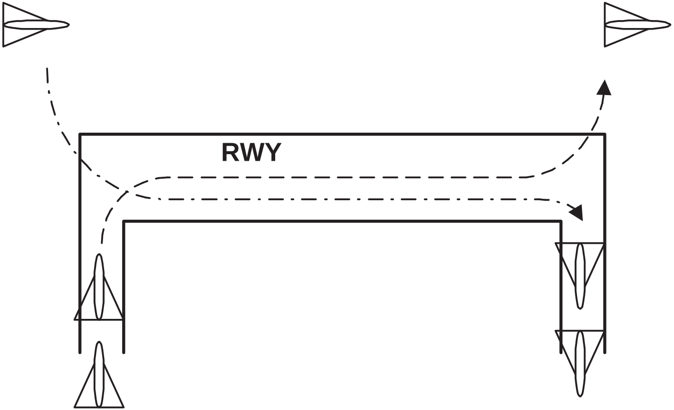
Automata - discrete event systems

	Automaton model	Discrete event state space model
State space	Q	$\mathcal{X} \in \mathbb{Z}^n$
Input u	string from Σ	discrete time discrete valued signal
Output y	string from Σ_O	discrete time discrete valued signal
State equation	$q(k+1) = \delta(q(k), u(k))$	$x(k+1) = \Psi(x(k), u(k))$
Output equation	$y(k) = \varphi(x(k))$	$y(k) = h(x(k), u(k))$

Introductory example: Garage gate



Simple example: Runway



Overview - Petri nets: modelling and dynamics

Petri net - abstract description: $\mathbf{PN} = (P, T, I, O)$

Static description (structure)

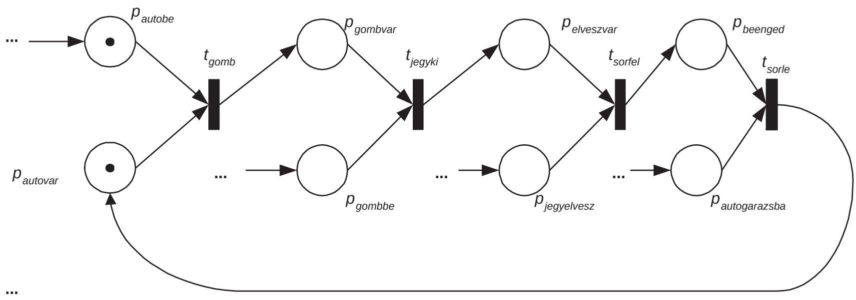
- set of **places (conditions)**: P
- set of **transitions (events)**: T
- **Input (pre-condition) function**: $I : T \to P^{\infty}$
- **Output (consequence) function**: $O : T \to P^{\infty}$

Graphical description: bipartite directed graph

- **Vertices**: places (P) and transitions (T) (partitions)
- **Edges**: input and output functions (I, O)

Example: garage gate – 1

Petri net model - graphical description



Example: garage gate – 2

Petri net model - formal description

Places (states; inputs):

$$P = \{p_{autovar}, p_{gombvar}, p_{elveszvar}, p_{beenged} \;;\; p_{autobe}, p_{gombbe}, p_{jegyelevesz}, p_{autoga}$$

Transitions:

$$T = \{t_{gomb}, t_{jegyki}, t_{sorfel}, t_{sorle}\}$$

Input function:

$$I(t_{gomb}) = \{p_{autobe}, p_{autovar}\} \quad , \quad I(t_{jegyki}) = \{p_{gombbe}, p_{gombvar}\}$$
$$I(t_{sorfel}) = \{p_{jegyelvesz}, p_{elveszvar}\} \quad , \quad I(t_{sorle}) = \{p_{beenged}, p_{autogarazsba}\}$$

Output function:

$$O(t_{gomb}) = \{p_{gombvar}\} \quad , \quad O(t_{jegyki}) = \{p_{elveszvar}\}$$
$$O(t_{sorfel}) = \{p_{beenged}\} \quad , \quad O(t_{sorle}) = \{p_{autovar}\}$$

Dynamics of Petri nets

Marking function: marking points (**tokens**)

$$\mu : \mathbf{P} \to \mathcal{N} \quad , \quad \mu(p_i) = \mu_i \geq 0$$
$$\underline{\mu}^T = [\mu_1, \mu_2, \dots, \mu_n] \quad , \quad n = |\mathbf{P}|$$

Transition **fires** (operates): when its pre-conditions are "true" (there is a **token** on its input places)

$$\underline{\mu}^{(i)}[t_j > \underline{\mu}^{(i+1)}$$

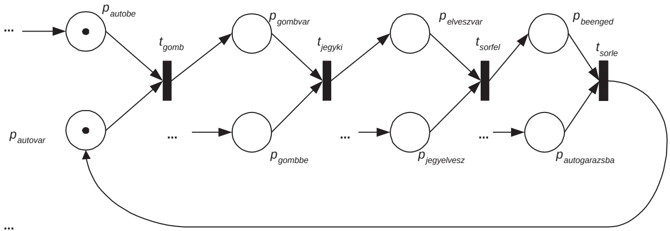
after firing the consequences become "true"

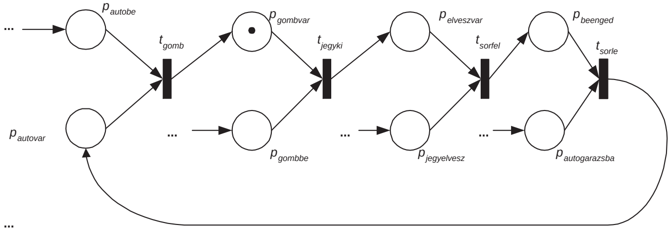
Firing (operation) sequence

$$\underline{\mu}^{(0)}[t_{j0} > \underline{\mu}^{(1)}[t_{j1} > ...[t_{jk} > \underline{\mu}^{(k+1)}$$

Example: garage gate – 3

One operation steps





Example: garage gate – 4

Formal description of an operation step

Marking vector

$$\underline{\mu}^T = [\mu_{autovar}, \mu_{gombvar}, \mu_{elveszvar}, \mu_{beenged} \; ; \; \mu_{autobe}, \mu_{gombbe}, \mu_{jegyelevesz}, \mu_{autogarazsba}]$$

Operation (firing) of transition t_{gomb}

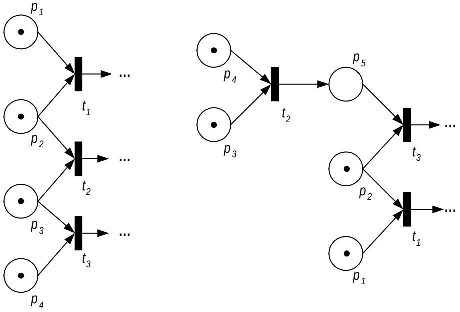
$$\underline{\mu}^{(1)}[t_{gomb} > \underline{\mu}^{(2)}$$

$$\underline{\mu}^{(1)} = [1,\ 0,\ 0,\ 0\ ;\ 1,\ 0,\ 0,\ 0]^T$$

$$\underline{\mu}^{(2)} = [0,\ 1,\ 0,\ 0\ ;\ 0,\ 0,\ 0,\ 0]^T$$

Parallel events

More than one enabled (fireable) transition:
concurrency (independent conditions), conflict, confusion



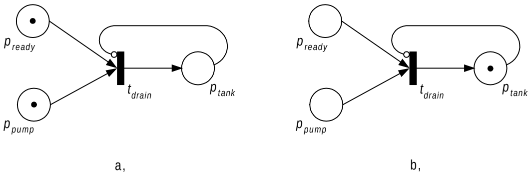
a,

b,

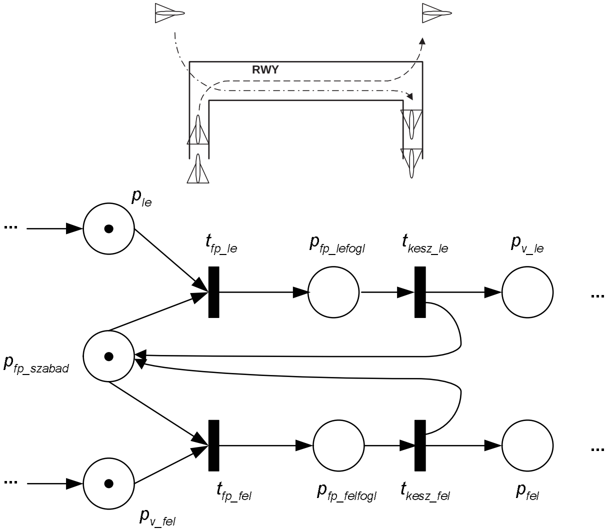
Conflict resolution

Using **inhibitor edges**:
- priority given by the user
- test edges

Other solutions:
- capacity of the places

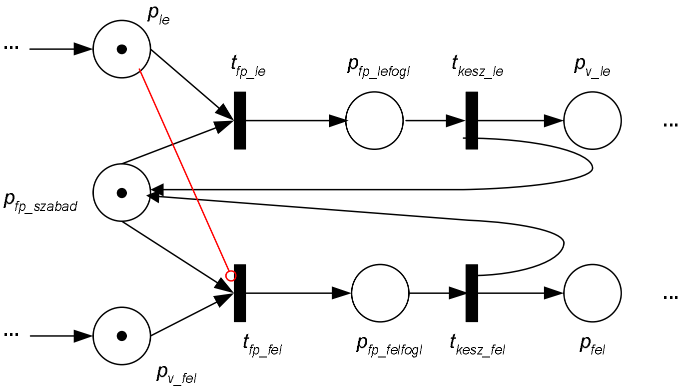


Petri net model of a runway – 1



Petri net model of a runway – 2

Conflict resolution: landing aircraft has priority



Overview - Solution of Petri net models

The solution problem

Abstract problem statement

Given:

- a *formal description* of a discrete event system model
- *initial state(s)*
- *external events*: system inputs

Compute:

- the sequence of *internal (state and output) events*

The solution is **algorithmic**! **The problem is NP-hard!**

Petri net models – reachability graph

Solution: marking (systems state) sequences
reachability graph (tree) (weighted directed graph)

- *vertices*: markings
- *edges*: if exists transition the firing of which connects them
- *edge weights*: the transition and the external events

Construction:

1. *start*: at the given initial state (marking)
2. *adding a new vertex*: by firing an enabled transition (with the effect of inputs!)

May be NP-hard (in conflict situation or non-finite operation)

The state space of Petri net models

State vector: marking in *internal* places
in- and out-degree is at least 1

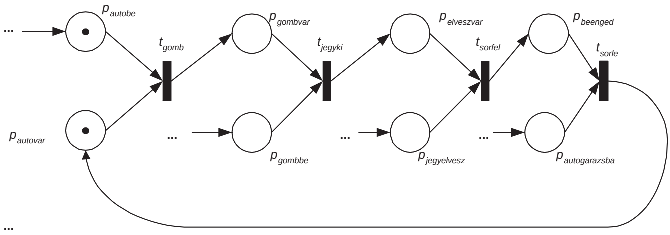
$$x(k) \quad \sim \quad \underline{\mu}_x^{(k)}$$

Inputs: marking in *input* places
in-degree is zero

$$u(k) \quad \sim \quad \underline{\mu}_u^{(k)}$$

Example: garage gate

Petri net model



$$\underline{\mu}_x^T = [\mu_{autovar}, \mu_{gombvar}, \mu_{elveszvar}, \mu_{beenged}]$$
$$\underline{\mu}_u^T = [\mu_{autobe}, \mu_{gombbe}, \mu_{jegyelevesz}, \mu_{autogarazsba}]$$